JOHANNES KEPLER
UNIVERSITÄT LINZ

Dr. Stefan Marr
Institute for System Software
stefan.marr@jku.at
Office S3 202

Master's Thesis

Fork/Join Work-Stealing For Truffle without Sequential Overhead

Student: Vaishali Dhanoa

Advisor: Stefan Marr, Daniele Bonetta

Start date: 15-Nov-2016

Fork/Join-based parallel programming is a versatile programming model, which combined with work-stealing scheduling allows for low-overhead dynamic load distribution. This makes it an excellent tool to parallelize workloads for which the structure depends on the input data or other dynamic characteristics. Unfortunately, in Java fork/join parallelism comes with a high sequential overhead, because even if a worker can process all tasks itself, the corresponding task objects need to be created and are potentially scheduled on the work-stealing queue. This means, programmers have the extra burden of introducing sequential cut-offs to ensure a task granularity that benefits from parallel execution. Moreover, scheduling and executing tasks in Java's Fork/Join executor might prevent compiler optimizations due to the Java Memory Model semantics.

In collaboration with Oracle Labs, we are investigating whether optimizations that avoid task creation [1] can be applied to Truffle-based AST interpreters [2]. The goal is to replicate experiments that have been done for custom JVMs based on the Truffle infrastructure. Furthermore, it needs to be investigated whether it is possible to use these or similar techniques to avoid overhead for fork/join task creation in peak-performance for parallel benchmarks.

The main research challenge seems to be to find techniques that can rely on the existing Truffle mechanisms for optimizations based on AST nodes, specialization of operations, speculative compilation, and deoptimization. The main challenge would be to devise a low-overhead technique for interaction between threads that does not require threads to fallback to interpreter execution after work was stolen from them.

The progress of the project should be discussed at least every two weeks with the advisors. A time schedule and a milestone plan must be set up within the first 3 weeks. It should be continuously refined and monitored to make sure that the thesis will be completed in time.

[1] Kumar, V.; Frampton, D.; Blackburn, S. M.; Grove, D. & Tardieu, O. (2012), Work-Stealing Without The Baggage, in Proc. of the 2012 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages & Applications (OOPSLA 2012), 2012, ACM, pp. 297-314.
[2] Würthinger, T.; Wöß, A.; Stadler, L.; Duboscq, G.; Simon, D. & Wimmer, C. (2012), Self-Optimizing AST Interpreters, in Proc. of the 8th Dynamic Languages Symposium, pp. 73-82.